A Novel Cluster-based Routing Protocol Optimization Approach for Wireless Sensor Mesh Networking

Javier Gordillo, Gabriel Mujica, Jorge Portilla, Teresa Riesgo
Centro de Electronica Industrial, Universidad Politecnica de Madrid
Email: {j.gordilloh, gabriel.mujica, jorge.portilla, teresa.riesgo}@upm.es

***Abstract*— In this work, a novel and powerful mechanism for optimizing dynamic routing protocols in wireless sensor mesh networking is deeply proposed and studied, by taking advantage and the benefits of flat-based routing techniques in combination with hierarchical strategies, so that a more efficient and energy/processing aware multi-path dissemination protocol is fully implemented. The communication and routing capabilities of WSN-based smart applications are key issues to be tackled in order to assure the reliability, scalability and long-term operability of the whole system. In order to accomplish such a challenging approach and targeting the on-site performance analysis of routing protocols in real scenarios, a new intelligent, mobile and adaptable routing protocol simulator is also proposed, so that users are provided with a complete comparative study of different multi-hop mesh-based network deployments based on the design constraints and application requirements.**

Keywords—WSN routing protocol, HW-SW co-design, sensor node deployment, mesh networking, cluster-based routing.

I. Introduction

The Wireless Sensor Network (WSN) research field has been undergoing an incredible technological growth in the last years due to the expansion of the Internet of Things (IoT) paradigm together with the inclusion of intelligent implementations for smart city applications [1], which leads to develop new hardware and software techniques to cope with the main requirements and constraints that these types of applications pose to technology designers. However, the creation and optimization of WSN-based systems goes beyond robust and low cost platforms, since the distribution, integration and cooperative processing of hundreds or even thousands of small devices is a critical aspect to assure the final operability and reliability of the planned application. Moreover, the release and autonomy of the sensor nodes is focused on the basis of self-network reconfiguration and dynamic behavior of the deployed devices, so as to bring the possibility of adapting their functionalities not only according to their local state (power consumption, processing elements, parameter and peripheral reconfiguration) but also in relation with the surrounding environment.

In this direction, the communication performance is a key aspect to establish an overall system correlation as a whole network more than the individual capability of deployed sensor nodes, so that the designed functionality of the application can be properly covered. Furthermore, network sensing and processing distribution hardly depends on the efficiency of the dissemination strategies to be used according to the deployment topology and partitioning [2]. Based on this, efficient multi-hop networking is an important research area in which different approaches have been proposed targeting the reliability and effectiveness of routing algorithms to communicate remote points [3]. Based on this, the problem of optimizing routing techniques for sensor networks has been addressed from various perspectives depending on how flexible and adaptable the communication is to be, specially in those situations where the connectivity between nodes is prone to change or present possible instability issues. Concerning the support of fully mesh deployments, two main approaches can be distinguished. First, flat-based routing protocols [4], where every node of the network has the same role so that the creation process of remote communication paths can be started, commanded and controlled by any device, and connections are not restricted to data packet retransmissions to centralized points or collecting nodes. Although flat-based routing protocols are focused on proactive or reactive mechanisms [5], the on-demand strategy to discovery dissemination paths could fit most of the WSN communication requirements. Two main contributions that have been widely used in the state of the art can be analyzed.

On one hand, the Dynamic Source Routing protocol (DSR) [6], relies the communication between two remote points in the inclusion the whole sequence identification of the nodes in the packet frame header along the path. On the other hand, in the Ad-hoc On-demand Distance Vector Routing protocol (AODV) [7] the routing discovery process is performed only when a data communication is requested. Such request is carried out by the source node, by broadcasting a query packet to its surrounding devices and then they continue disseminating the request action. Unlike DSR, in this scenario the source node only has information of the next hop of the path, so the packet frames are considerably reduced. Although flat-based routing protocols such as the AODV algorithm provide a very flexible and adaptable alternative to wireless mesh networking, in large-scale deployment distributions it may be more suitable to split it into several sub-networks or groups trying to avoid flooding the overall network with control packets, which can lead to decrease the performance of the dissemination process and then penalize the power consumption and autonomy of the sensor nodes.

This is the main reason why hierarchical routing protocols appear as an alternative to create a structural topology based on assigning different functional roles to the sensor nodes according to specific requirements. One of the well-known mechanisms within this approach are the clustering protocols [8], where routing capabilities relies on the creation of cluster nodes that are in charge of conducting the data retransmissions among different deployment areas or groups. Taking advantage of this approach and also combining the main benefits of flat-based routing protocols, in this work a novel and powerful routing protocol optimization technique for wireless sensor networks is fully proposed to provide deployers with a real and dynamic multi-hop strategy to distribute sensing and processing information in a mesh scenario, targeting its flexibility, scalability and the applicability to large-scale deployment scenarios.

The design and implementation of a cluster-based routing protocol optimization mechanism is focused on its inclusion into two main platform: the first one a modular hardware-software WSN platform called Cookie node [9], which integrates in a unique architecture a set of processing, communication and sensing capabilities; and the second one a novel in-field routing simulation tool that is also proposed in this work to tackle the main lack of well-defined modeling frameworks to carry out the analysis of multi-path mesh networking in-situ. Although there are different simulators in the state of the art for routing protocols, such as [10][11], few of them try to provide a runtime on-site comparative analysis of several routing implementations so that users can have a better understanding on how the performance of the wireless deployment can be affected by the routing mechanism to be adopted or, instead, how the specific constraints of the deployment scenarios may lead to decide whether a routing protocol can be more appropriate for the efficiency and long-term operability of the system in a particular scenario.

The rest of the paper is organized as follows, starting in section 2 with a deep study of the proposed cluster-based routing protocol together with the main optimization techniques that have been designed. In section 3 the implementation of the routing mechanism is detailed by presenting the proposed simulation tool. In section 4 the experimental results are fully analyzed whereas in section 5 conclusions and main contributions are provided.

II. Cluster-Based Routing Protocol – Proposed Technique

One of the main issues that is often tackled in flat-based routing protocols under MANETs is its performance for large network sizes due to link overhead. The solution applied in many cases for this scalability problem is establishing a hierarchy in the wireless network. Cluster-based routing protocols divide geographically the network into groups (clusters) and assign a functional role or status to each node in the protocol mechanism. In this way, the information of each region is focused on some nodes, called cluster-heads, which are responsible of ensuring communications within their domain. Therefore, the overall deployment information is distributed among key nodes that collaborate to disseminate the corresponding data between remote points in a multi-hop strategy network.

Based on this approach, many of the cluster-based routing protocols split the network into clusters according to the planning stage results prior to the deployment phase. These protocols, however, may not be always recommended for MANETs, due to the instability of communications, the variability of the links and thus the need for hierarchy changes. Other protocols establish cluster-heads in a flat-based fashion by using a stochastic algorithm [12] or by comparing the nodes' height with the corresponding neighbors according to their identification number [13]. Generally, in any of these protocols, in case of adding new nodes to the network deployment, they would usually try to find a cluster-head in their neighborhood and then get attached to it. If failed, they would become cluster-heads themselves.

The proposed protocol has been designed as clear optimization mechanism for flat-based network scenarios, by establishing the network hierarchy creation rules based on the aforementioned requirements. In this direction, the parameter *weight* is introduced in cluster-based protocols such as the one proposed in [14]. The weight of a node is a weighted expression that serves as a classification element and can include parameters such as the number of neighbors and/or the consumption level of the radio communication module. The higher the weight is, the more possibilities a node has to become a cluster-head. Therefore, the proposed routing protocol adopts a weight-based distribution approach by selecting fittest nodes to become cluster-heads regardless of their identification or coordinates.

The structure of the proposed cluster-based routing protocol distinguishes among 4 possible node's status or functional roles, as shown in Figure 1:

- ***Isolated***: the node is not included in the hierarchy. Although it may be aware of its neighbors, it has not been included into the cluster-based system. This is the default initial status for the nodes of the network.
- ***Normal***: the node is part of a cluster defined by a cluster-head node. It is then subordinated to its cluster-head, and will request further information to it when needed.
- ***Gateway***: the node may be both part of a cluster or part of two different clusters. It is responsible for the communication between cluster-heads and serves as a highway of information exchanging.
- ***Cluster-head***: the node is in charge of its corresponding cluster. It generally has a series of normal nodes under its domain, and manages advanced procedures within the cluster regarding the multi-hop communication mechanism.

A. Protocol information Management

Regardless of the node functional roles, every node has two different protocol information tables: the *Neighbor Table* and the *Routing Table*. The *Neighbor Table* of a node contains information about every node within its range of communication. This information includes its id, node status, its cluster-head, weight, a sequence number, and the cost involved in the communication (for instance, if the cost is assigned to the number of hops relation, it will not be necessary because in this case it will always be 1). Meanwhile, the *Routing Table* is similar to the one used in the AODV protocol: once a route has been discovered, the id of the destination node is registered, along with the 'next hop' node, the sequence number and the cost of a route.

Apart from this scheme, *Cluster-head nodes* have two additional tables to support the cluster-based structure of the system:

- The *Cluster-head Neighbor Table* includes the id of neighbor cluster-heads, and the immediate gateway node that heads to it. This gateway is similar to the 'next-hop' node in the Routing Table. Sequence number and cost are also saved.

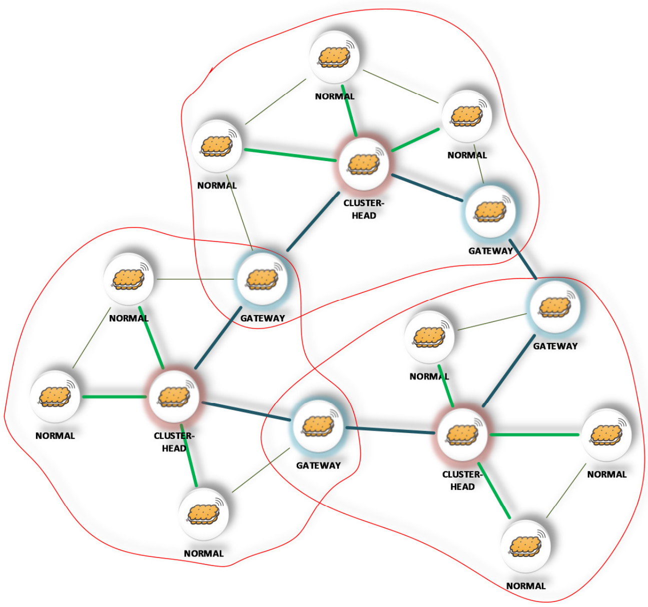

Fig. 1: Node's functional roles in a cluster-based structure.

• The *Cluster Table* includes the id of the normal nodes that belong to the cluster. Moreover, their own neighbors and its cluster-heads are registered as well. By registering this information schema, the cluster-head is fully and directly aware of nodes 2-hops ahead from it. Sequence number and cost of each hop are also computed and recorded in the corresponding table entry.

It is important to highlight that the information on nodes 2-hops away is already used in the cluster-based routing protocol proposed in [14], although they define such information to be kept in every node of the network, whereas in the protocol proposed in this work only cluster-head nodes maintain it, which lead to a big percentage of data packages saving (and thus reducing overall power consumption and network overhead), while cluster-heads behave as intelligent managers of the network communication and routing capabilities.

Gateway nodes have also a special additional table called *Gateway Table*. It registers every pair of cluster-heads that use that gateway to communicate to remote deployment areas. The table has fields for the ids of those cluster-heads and for the 'next-hop' nodes needed for sending data to them. This 'next-hop' nodes may be other gateways or the cluster-heads themselves.

Once every type of node has been defined with their functional role and information table registering, the cluster formation process together with the routing protocol itself and the local repair procedure are described as follows.

B. Cluster Formation

Based on the cluster formation process proposed by authors in [15], some important modifications have been designed and applied in this work in order to optimize the algorithm and enhance the performance of the routing mechanism, specially targeting the energy and processing constraints of wireless sensor nodes.

According to this algorithm, in a first stage every node in the network launches a limited number of broadcast messages with basic information: id, status, weight parameter and cluster-head. These messages are usually called *hello* or *live messages* and will be received by nearby nodes, which will use the incoming data to create new entries in their *Neighbor Tables*. Although only one *live message* would be needed for this purpose, the reason why several attempts are made is due to the instability nature of communications in WSN-based deployments.

Once this first stage of recognition is over, each node checks with its *Neighbor Table* whether or not it is the best node of the neighborhood, according to a comparative analysis of the weight parameter. If this is the case, it sets their status to cluster-head, and notifies their neighbors via *live message* broadcast. In this case, as the node has information related to its neighbors, there is no need to subsequently repeat the message if a link-based acknowledgement mechanism is activated. Nearby, nodes will receive the message and three possible cases will be triggered:

• If the neighbor node has an isolated status, this will become a *normal node* and set its neighbor as its *cluster-head*. Information about its neighborhood will be sent to its new cluster-head, which will save it in the corresponding *Cluster Table*.

• If the neighbor node is another *cluster-head*, this will use data received to create a new entry in its *Cluster-head Neighbor Table*. Then, the source cluster-head node will set itself as gateway in the table.

• If the neighbor node is a *normal node* that has a different cluster-head (therefore, it belongs to another cluster), this node will send its cluster-head a gateway-request message.

A *gateway-request message* contains information related to the new *cluster-head* and the path to reach it. Once a cluster-head receives a *gateway-request message* from one of its subordinate nodes, it checks whether a route to a new cluster-head has been discovered; and if not, whether the new route is better in terms of cost than the one registered. If a better route is already registered, the subordinate node will receive a negative response, called *NRgateway message* with no effects. However, if a new route to the cluster-head is added, an affirmative response, called *ARgateway message*, is sent back to the subordinate node, which will become a gateway node. This gateway will add the route between cluster-heads to its Gateway Table, and forward the *ARgateway message* to the other neighbor cluster-head. Thereby, this data will serve the cluster-head to update *its Cluster-head Neighbor Table*.

A similar process is carried out when a gateway node receives a live message from a cluster-head that has not been registered in the *Gateway Table*. In that case, a gateway-request message is sent to every registered cluster-head, and every received *ARgateway message* is then forwarded to the new cluster-head.

In order to keep information updated of the node status to assure the integrity of the multi-hop communication, as soon as a node changes its functional role, it must broadcast a live message to notify its neighbors and then wait for an acknowledgement response from all of them.

Because of this update, a normal node could contact to another normal node with a different cluster-head. The behavior is the same as if it had received a live message from the cluster-head of its neighbor; however, a new parameter is added both in the gateway-request and *ARgateway messages*: the '*intermediate gateway*' with the id of the other normal node. Both normal nodes will then become gateways and they will mutually add each other and their cluster-head to each corresponding *Gateway Table*.

This will also happen between two gateways or between a normal node and a gateway. In such a case, gateway nodes will check in *their Gateway Tables* for the cluster-head of the other node rather than checking its own cluster-head.

If an isolated node contacts a normal or gateway node, it does not change its status. Therefore, after a certain amount of time, if a node remains isolated, it checks its *Neighbor Table* and sends a *CHRequest message* to its best neighbor in terms of weight. When the neighbor node receives the message, it automatically becomes a cluster-head and notifies its neighborhood. Then, the isolated node becomes a normal node and sets its neighbor as its cluster-head. Once there are no isolated nodes left, the cluster forming process is finished.

As a summary, in Table 1 the protocol responses for every possible *live message* are shown in detail.

C. Route Discovery Protocol

Once clusters have been formed and there are no isolated nodes left, communication between nodes is made on-demand. As AODV does, each node keeps information of the next-hop in the route to a destination node. In this sense, the performance of the routing mechanism is quite similar to AODV. The main difference is that, while AODV runs on a flat-based network and uses broadcast messages for route discovering, the proposed protocol uses hierarchy to delegate on cluster-heads such responsibility. By doing this, there is no need for broadcast messages beyond neighbor discovering, which has a direct effect on a better performance in terms of reducing power consumption, data processing elements and network overhead.

In the protocol proposed in this work, when a node requires sending a message to another one, it checks its *Neighbor Table* and its *Routing Table*. If a route is found, the message is delivered to the next-hop node in the route or the destination node itself. If the destination node is not among the neighbor nodes and there is no evidence of previous communication with it, the discovering procedure begins.

Instead of sending broadcast messages, the source node sends a RREQ message to its cluster-head. If the source node is a gateway node, it will send the request to every cluster-head registered in its *Gateway Table*. A cluster-head that receives a RREQ message checks both its neighborhood and its Routing Table for the destination node. If it does not find it, it forwards the RREQ message to its neighbor cluster-heads through the corresponding gateway. This gateway also checks its *Neighbor Table* and *Routing Table* before forwarding the message to the cluster-head. As in the AODV protocol, each node that receives a RREQ generates the corresponding message including its Id, while sequence number prevents closed loops in the network. In addition, the TTL parameter strengthens this purpose. When a cluster-head discovers the destination node in its neighborhood or in its *Routing Table* (or the destination node itself receives the RREQ message), it sends back a RREP message to the previous cluster-head.

TABLE I. PROTOCOL INTERACTIONS FOR EVERY PARTICULAR NODE STATUS

Emitter Node status	Receiver Node status	Emitter Node CH	Receiver Node CH	Receiver Node Response
Isolated	Isolated	-	-	-
Isolated	Normal	-	CH-x	-
Isolated	Gateway	-	CH-x	-
Isolated	Cluster-head	-	R-Node	-
Normal	Isolated	CH-x	-	-
Normal	Normal	CH-x	CH-x	-
Normal	Normal	CH-x	CH-y	Gateway request to CH-y
Normal	Gateway	CH-x	CH-x	-
Normal	Gateway	CH-x	CH-y	Gateway request to CH-y if CH-x is not registered in Gateway Table
Normal	Cluster-head	R-Node	R-Node	-
Normal	Cluster-head	CH-x	R-Node	-
Gateway	Isolated	CH-x	-	-
Gateway	Normal	CH-x	CH-x	-
Gateway	Normal	CH-x	CH-y	Gateway request to CH-y
Gateway	Gateway	CH-x	CH-x	-
Gateway	Gateway	CH-x	CH-y	Gateway request to CH-y if CH-x is not registered in Gateway Table
Gateway	Cluster-head	R-Node	R-Node	-
Gateway	Cluster-head	CH-x	R-Node	-
Cluster-head	Isolated	E-Node	-	Becomes Normal node and subordinates to E-Node
Cluster-head	Normal	E-Node	E-Node	-
Cluster-head	Normal	E-Node	CH-x	Gateway request to CH-x
Cluster-head	Gateway	E-Node	E-Node	-
Cluster-head	Gateway	E-Node	CH-x	Gateway request to CH-x if E-Node is not registered in Gateway Table
Cluster-head	Cluster-head	E-Node	R-Node	C-H Neighbor Table update

The RREP message will, at first, travel its way back following the route that was obtained during the RREQ message dissemination procedure. Again, as in the AODV protocol, every intermediate node that receives a RREP message adds both the destination node and the source node to its own *Routing Table*.

However, while RREP is travelling back to the source node, cluster-heads can perform an optimization algorithm by trying to design an alternative route thanks to its Cluster Table. If succeed, the cluster-head will notify the nodes involved, add the alternative route to the RREP message and let the next cluster-head continue developing it from the optimization process already done. When the last cluster-head receives the RREP message, it checks whether the alternative route is more suitable or not in terms of the multi-hop communication cost metric and sends the best solution to the source node, as shown in Fig. 2.

Once a route has been discovered, the sent data follows the best route to its destination by travelling each time to the 'next-hop' node registered in each *Routing Table*.

D. Route Maintenance

In order to verify the integrity of the routing performance between remote points and assure the multi-hop communication in the proposed protocol, two different types of failures can be defined:

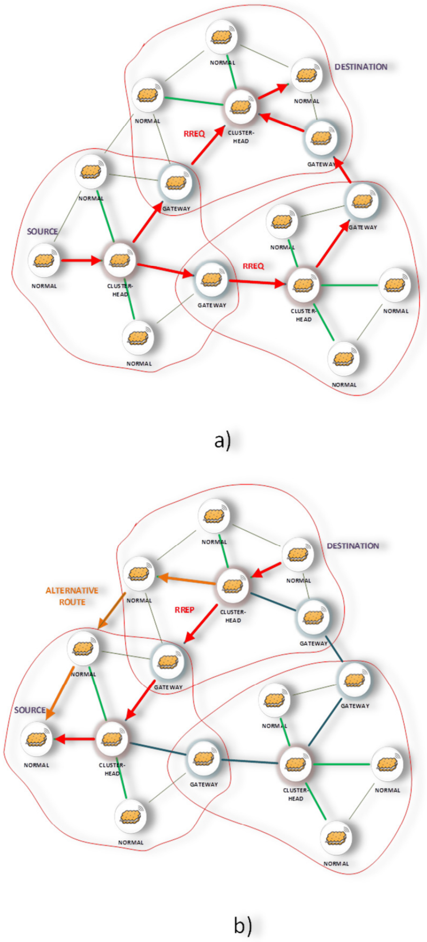

Fig. 2: Route discovering. a) RREQ travelling among cluster-heads b) RREP is sent back; although an alternative route is proposed.

• A failure in a main path: the error is detected in the path that goes from a cluster-head to another cluster-head or from a cluster-head to any of its subordinates.

• A failure in a secondary path: the error is detected between two normal nodes or between a gateway and a normal node. In a particular case, it might be also be found between two gateways from different paths.

When the main failure is found in the path between cluster-heads, an alternative path can be found just as the route discovering protocol: forwarding a message from cluster-head to cluster-head. In addition, a cluster-head can use its information about 2-hops away nodes and try to locate an alternative gateway route. If a new route is found, the old path between cluster-head is dismissed, and the gateways involved remove the path in their *Gateway Tables*, and they become normal nodes if necessary. However, if the new route is not suitable according to certain criteria (such as cost overrun), both cluster-heads involved may require all of their subordinates to send a *live messages* in order to find a new gateway if possible. Only if no alternative way is found to reach the cluster-head, the solution will be to alert the source node and then reorganize the network structure, launching again the cluster formation procedure.

In case the failure takes place between a cluster-head and its subordinate, the cluster-head would look for the lost node in its *Cluster Table* at a distance of 2 hops. If it is found, messages addressed to the lost node will be forwarded to the most suitable intermediate node in the cluster. If not found, it is likely that the node is isolated from the network.

Secondary failures are much easier to repair. If an error in communications is located between two normal nodes, the normal node will request its cluster-head an alternative path. The *cluster-head* will try to find an alternative path looking on its *Cluster Table*, and if it failed to find one, the cluster-head will forward a message to its neighbor cluster-heads. If the failure is detected by a gateway, a request to several cluster-heads can be done at a time. Both type of failures and the proposed protocol local repair procedures are shown in Fig. 3.

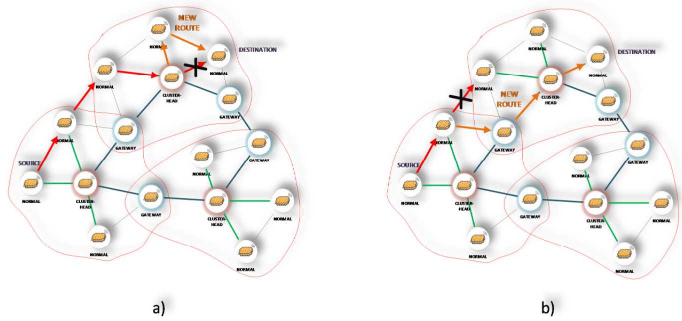

Fig. 3: Local Repair. a) Failure on main path. b) Failure on secondary path.

III. System Implementation

Along with the development of the proposed routing protocol, a simulation tool has also been implemented and is presented in this work. The aim of this tool is to simulate the behavior of multi-hop routing protocols and compare results among them targeting their inclusion into real application scenarios, so that a deeper analysis on how to select the most appropriate mechanism according the system requirements can be fully produced. In this way, the proposed simulation tool is focused on its on-site usability, that is, provide users with a powerful and dynamic tool to study the network behavior of the sensor deployment, in-field and in real-time. Moreover, this powerful tool is designed to provide a complete assess of the implemented routing algorithms into the Cookie platform, including the cluster-based protocol proposed in this work.

The main challenge of the tool, called Cookie-based Communication & Routing Simulation System (CB-RSim) is designing a simulation system where timing and synchronization are keys to success of the tool, together with an appropriate modeling of the node's functionality in terms of communication capabilities and the ease of integration in an in-situ commissioning toolset to analyze and validate de behavior of sensor network deployments. This required a thread-based schema for a more efficient system implementation. However, as CB-RSim is designed for Android-based devices instead of more powerful computers, hardware limitations must be considered. To meet these requirements, every action and event that will take place on the node simulation model is added to a list or runtime executable tasks, along with the planned time to be performed. The thread-based schema will continuously check the list of executable tasks and perform pending actions according to the time set and the own timing of the thread. According to this standard, activities such as sending a message to a node will require the source node to add the action into the executable task schema, and when that action is carried out, the destination node will introduce into the list the new action of receiving the message, within a time that models the communications and processing capabilities of the target nodes.

Despite the complexity of the internal process, this simulator for Android-based smart devices offers a friendly user interface, as seen in Fig. 4, aiming to provide user with an in-field easy-to-use and runtime evaluation of the WSN-based deployment scenario in terms of communication and routing functionalities. The interface is running upon the Google Maps capabilities so that most options available on that API are also available on the developed app (such as zooming, tilting, moving and changing the map type). New nodes can be easily included and translated, and every node added or dragged is saved in a local deployment database to avoid rebuilding the network each time the simulator is launched.

On the other hand, regarding the deployment simulation capabilities, neighborhood between nodes is marked by a green line joining both nodes. As no predefined network is loaded, the criteria to establish neighborhood between nodes is a ‘communications range’; that can be pre-established as an input data of the system based on the radio communication module or hardware node parametrization to be used. This range can be shown on the map as a circle around each node in order to help users to build the network and its connectivity, so that the communication performance can be evaluated and then include the corresponding behavioral model into the simulation system.

The user can also access a list of available protocol implementations and select one of them. From then on, every action will be performed according to the selected routing strategy. All the updated information that a node handles is displayed when selecting it. This is a key aspect offered by CB-RSim: it differentiates between information related to a node, and the information handled by it. For instance, a node may have several neighbor nodes, but as soon as no communication has been established, the node will not register related information about them and its *Neighbor Table* will be kept cleared.

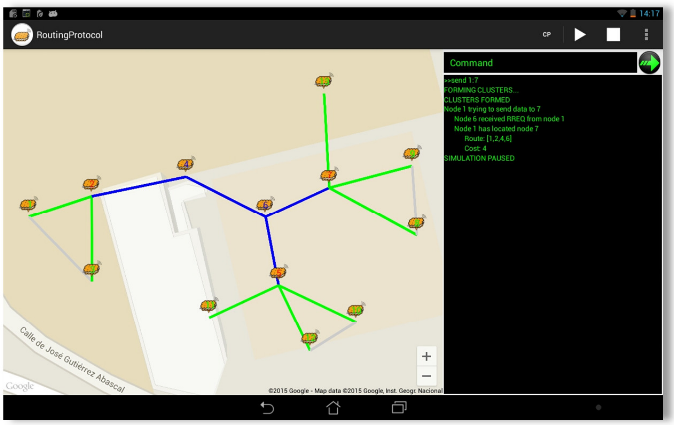

Fig. 4: CB-RSim Simulator implementation.

Although all simulations share the same mechanism to run the communication capabilities, two types of simulation can be distinguished:

- A discreet simulation: the simulation involves a communication between two specific remote nodes according to the selected protocol. Once the message has been received according the corresponding routing strategy, it finishes.
- A continuous simulation: the simulation can only be stopped by the user. Periodically, two random and different nodes establish a communication according to the selected protocol.

In both cases, for routing strategies such as the proposed Cluster-based protocol, procedures such as cluster formation tasks will be done on-demand before establishing communications, when needed.

CB-RSIm provides user with a dynamic console that logs every relevant event or action during the simulation. When the simulation is over, a series of parameters regarding the performance of the selected routing protocol are shown, such as the total number of broadcasts or messages delivery rate, so that a comparison between different protocols can be made. This console is also provided with an input instruction set, where several commands can be written for advanced options, such as connectivity breaking to enforce failure or establishing certain communication ranges.

Currently, two protocols have been implemented into CB-RSim: the Cluster-Based Routing Protocol proposed in this work and the AODV Routing Protocol. The main objective is to compare both algorithms, and from the results obtained by CB-RSim, redesign those aspects of the proposed protocol where there is room for improvement. Furthermore, simulations can be run for different types of network topologies and size, targeting scalability of the system, which also make it easier to locate the strengths and weaknesses of each protocol. This advantage has been used in order to improve the proposed protocol as it was initially implemented.

In addition to this, one advantage of programming for Android devices is the ease of embedding a developed code into a bigger implementation. Actually, this is the purpose of the CB-RSim simulator tool: to be part of a more complex tool that has been designed for commissioning and maintenance tasks of WSN-based smart systems [16]. In this way, the user would be able to simulate several algorithms for the deployed network, adapting the routing protocol to the specific requirement of the target WSN application scenario. The support tool would be able to include and reconfigure the selected protocol into the real nodes to be used in-field.

IV. EXPERIMENTAL RESULTS & TEST CASES

In order to analyze the behavior of the proposed protocol and compare it to the AODV routing strategy by using CB-RSim, a test case scenario has been proposed. This experimental case involves 30 nodes forming the flat-based network shown in Fig. 5. Every communication between nodes is considered bidirectional and each node has a unique id. Neighborhood is represented by a thin green line; however, nodes are not aware of their neighbors until communications have been properly established.

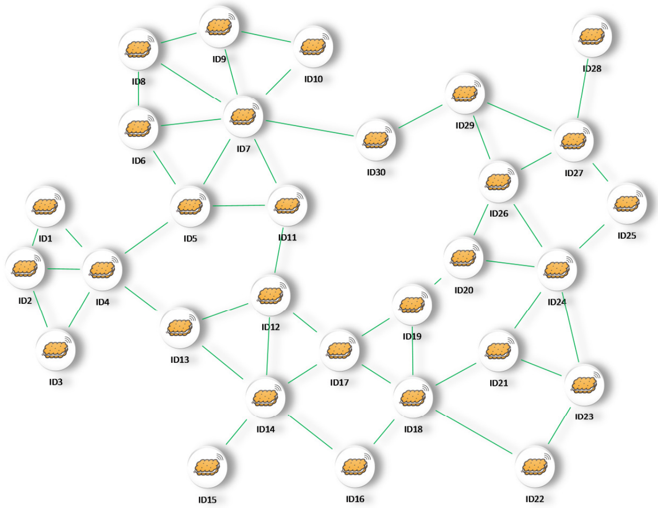

Fig. 5: WSN experimental test case.

For both protocol implementations, route discovering procedures have been analyzed. For instance, a communication request such a X>Y implies that node X sends a RREQ message to the corresponding subsequent points in order to find node Y. The route from node X to node Y that passes through node k is represented as 'X-…-k-…-Y'. If the RREP response is generated by node f, which has found node Y in its *Neighbor Table* or in its *Routing Table*, the route is represented as 'X-…-k-…-[f]-…-Y'. The main parameters that are taken into account to carry out the comparative analysis of the algorithms are the energy consumption of a routing path, the cost of discovering and maintaining the communication between two remote points and the transmission delay among the network connections, which are computed as the involved broadcasts of the selected mechanism, the total amount of broadcast spent and the number of hops associated to the discovery paths.

A. AODV Protocol

In the original AODV protocol, if node X that needs to contact node Y, looks on its *Neighbor Table* or its *Routing Table*. If not found, it sends a broadcast message (RREQ) to any of the surrounding nodes that can listen to it. If a node receives a RREQ message, replies with an acknowledge message, adds the source node to its Neighbor Table if necessary, and looks for node Y on its tables. If it does not succeed, it also sends a broadcast message to its neighborhood.

That way, while node X is looking for node Y, all nodes involved in the process are discovering their own neighbors if they have not done it before. Once a node has located node Y or node Y itself has received the RREQ message, an RREP message is sent back to node X through the route followed by the RREQ message. In Table 2, a list of performed requests that have been executed by CB-RSim is shown in detail, with the corresponding cost results when applying the AODV routing protocol.

The original AODV mechanism is widely used in MANETs, and it strength underlies in its full adaptability, flexibility and scalability to different mesh networking scenarios. However, as shown in the experimental results of Table 2, one of its weaknesses in its origin concept is the fact that may flood the network with broadcast messages when remote node requires finding another one in large scale and crowded situations. This is one of the key points that the proposed protocol tries to address in order to optimize the dynamic behavior of the routing algorithm.

TABLE II. EXPERIMENTAL RESULTS OF CB-RSIM FOR AODV.

Request	Route found	Number of hops	Broadcasts involved	Total Broadcasts
3>28	3-4-5-7-30-29-27-[28]	7	29	29
8>22	8-7-11-12-17-[18]-22	6	25	54
25>15	25-24-21-18-17-[14]-15	6	27	81
24>3	24-26-[29]-30-7-5-4-3	7	15	96
11>20	11-12-17-19-20	4	24	120
14>7	14-12-[11]-7	3	20	140
11>3	11-[5]-4-3	3	15	155
20>22	20-24-[23]-22	3	9	164
16>29	16-18-19-20-[26]-29	5	24	188
26>10	26-29-30-[7]-10	4	25	213
6>24	6-[7]-30-29-26-24	5	4	217
Local Repair: 12>17	12-14-17	2	24	241

B. Proposed Cluster-Based Routing Protocol

One of the aims to establishing a hierarchy in the network is to avoid the high rate of broadcast messages present in some flat-based mechanisms. However, in the proposed protocol, a first stage of broadcast messages is needed for that purpose. As explained in the Cluster Formation section, each node has to send a limited number of broadcast messages to discover its neighbors. Furthermore, when a node changes its status, it also sends a broadcast message alerting its neighbors of such configuration.

For the proposed experimental network, if the limit of broadcast messages that each node can send for discovering its neighbors is set to 5 (a extreme case), a total of 150 messages would be sent. For every node that changes its status, one should be added, performing around 180 messages. Some of the nodes will change its status again (like normal nodes that become gateways), increasing up to 200 the number of broadcast messages sent. The result of the process can be seen in Fig. 6, where clusters are delimited by a red line, cluster-head nodes are shown in red while gateways are in blue. Main routes are shown wider and in blue if joining cluster-heads, or in green if connecting a normal node with its cluster-head.

Although this may seem to be a high rate of broadcast messages; actually these will be the only control message actions to be performed when the organization of the network is required. Therefore, there is no unnecessary network flooding during runtime communication execution of the wireless sensor nodes. Table 3 shows the results for the proposed protocol mechanism on the same network and with the same requests than in the AODV experimental case. The route represented is the one followed by the RREQ message. Bold sections in the route represent sections of a main path.

However, according to the proposed protocol, the route created for connecting two nodes is not necessarily the same followed by the RREQ message. If the cluster-heads includes the proposed optimization algorithm for the returning process, alternative routes may be discovered, which are displayed in brackets.

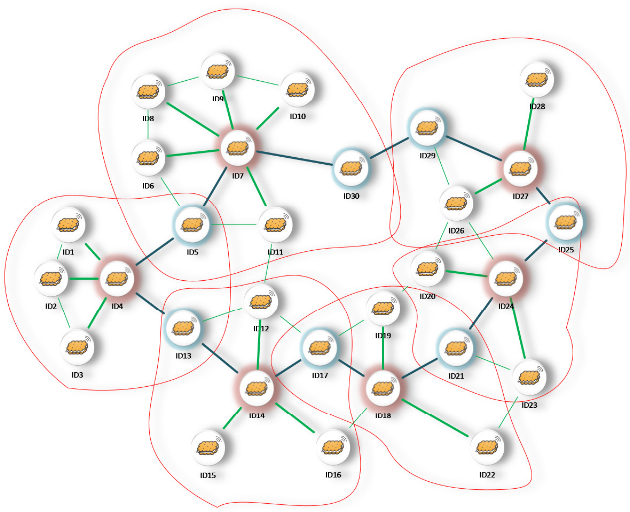

Fig. 6: Cluster formation in the proposed experimental test case.

C. *Comparative Analysis*

As it can be seen in the experimental results, for the AODV protocol, the number of total broadcast may increase rapidly with each route discovering process, although this is also dependent on the network distribution as well as the instability of the deployment. For the proposed scenario, the number of total broadcast messages reaches the level at the cluster formation process when requesting 10 different discovery routes. While this number increases almost proportional to the network size in the proposed cluster-based protocol, in case of the AODV tends to be factorial. However, it is not only the number of these types of messages what can affect the performance of the system, but it is also the fact of flooding the network. Therefore, it is clear from the experimental results and the discovery mechanism itself that this is the main strength of the proposed protocol: while using a flat-based network, it establishes a smart hierarchy where some nodes are in charge of control packages related to the routing discovery. That way, the rest of the network deployment can be reached by other nodes with a minimum cost.

However, it is true that this fact is affected by the topology of the network. For instance, in the extreme case of having all the nodes lined up in a row, much fewer broadcasts would be needed, as it would be much easier to find a node that is part of a route to a certain node. The opposite case, a fully mess-type network, would surely be much more suitable for a cluster-based protocol, as AODV broadcasts would spread along the network. The same advantage is used when repairing locally this type of network: while AODV uses in its basic operation broadcast messages to find an alternative route, the cluster-based protocol applies cluster-head information about its surroundings and easily finds a solution to forward the data packets.

The results obtained in terms of number of hops could be marked as one of the key points of the protocol to be enhanced, although the number of neighbors' discovery broadcasts has been set higher. Nevertheless, it has been shown that a cluster-based routing protocol with smart cluster-heads can reach results as good as the AODV protocol. These optimization algorithms, although not deeply detailed, can be generated just with the information related to the nodes 2-hops away and its corresponding cluster-heads, so that alternative route can be easily proposed.

TABLE III. EXPERIMENTAL RESULTS OF CB-RSIM FOR CLUSTER-BASED PROTOCOL.

Request	Route found	Number of hops	Broadcasts involved	Total Broadcasts
Cluster formation	-	-	200	200
3>28	3-**4-5-7-30-29-[27]**-28 (3-4-5-7-30-29-27-28)	7 (7)	0	200
8>22	8-**7-5-4-13-14-17-[18]**-22 (8-7-11-12-17-18-22)	8 (6)	0	200
25>15	25-**24-21-18-17-[14]**-15 (25-24-21-18-16-14-15)	6 (6)	0	200
24>3	**24-21-18-17-14-13-[4]**-3 (24-20-19-17-12-13-4-3)	7 (7)	0	200
11>20	11-**7-30-29-27-25-[24]**-20 (11-7-30-29-26-20)	7 (5)	0	200
14>7	**14-13-4-5-[7]** (14-12-11-7)	4 (3)	0	200
11>3	11-**[7]-5-4**-3 (11-5-4-3)	4 (3)	0	200
20>22	20-**24-21-[18]**-22 (20-24-23-22)	4 (3)	0	200
16>29	16-**14-13-4-5-7-30-[29]** (16-18-21-24-26-29)*	7 (5)	0	200
26>10	26-**27-29-30-[7]**-10 (26-29-30-7-10)	5 (4)	0	200
6>24	6-**7-30-29-27-25-[24]** (6-7-30-29-26-24)	6 (5)	0	200
Local Repair: 12>17	12-14-17	2	0	200

*In this case, when applying the returning optimization algorithm, the RREQ route would be 16-14-17-18-21-24-25-[27]-29.

V. CONCLUSIONS & CONTRIBUTIONS

In this work a very flexible and adaptive routing protocol for multi-hop wireless sensor networks has been presented, taking advantage of the main capabilities of flat-based routing mechanism in combination with the efficiency of a cluster-based optimization strategy. According to the experimental results and the comparative analysis of both types of approaches, the instability of the network and the level of link volatility will play a key role to decide which mechanism is more suitable to be applied in particular experimental scenarios, so the proposed CB-RSim provides a powerful approach for users to have a better understanding of the wireless sensor network and its reconfigurability according to the expected functional behavior and system performance. This is a very important step in the integration of novel routing protocol optimization techniques within the proposed simulator tool to foster their inclusion into the modular implementation of the Cookies development framework, which attempts to provide user with a complete deployment toolkit to evaluate and validate WSN smart applications based on this growing technology.

REFERENCES

[1] Cardone, G.; Bellavista, P.; Corradi, A.; Foschini, L., "*Effective collaborative monitoring in smart cities: Converging MANET and WSN for fast data collection,*" Proceedings of ITU Kaleidoscope 2011: The Fully Networked Human? -

Innovations for Future Networks and Services (K-2011), pp.1,8, 12-14 Dec. 2011.

[2] Wenxiang Li; Yunhe Wu; Chunsheng Zhu; Yajie Ma, "*Performance Comparison of Source Routing Tactics for WSN of Grid Topology,*" IEEE 12th International Conference on Dependable, Autonomic and Secure Computing (DASC), 2014pp.295, 300, Aug. 2014.

[3] Kaur, B.; Kaushal, S., "*QoS based evaluation of routing protocols in WSN,*" Recent Advances in Engineering and Computational Sciences (RAECS), 2014, pp.1,7, March 2014.

[4] Kanavalli, A.; Sserubiri, D.; Deepa Shenoy, P.; Venugopal, K.R.; Patnaik, L.M., "*A flat routing protocol for sensor networks,*" Proceeding of International Conference on Methods and Models in Computer Science, 2009. ICM2CS 2009, pp.1-5, Dec. 2009.

[5] Khushboo Tripathi, Manjusha Pandey, and Shekhar Verma. "*Comparison of reactive and proactive routing protocols for different mobility conditions in WSN*" In Proceedings of the 2011 International Conference on Communication, Computing & Security (ICCCS '11). ACM, New York, NY, USA, 156-161, 2011.

[6] Aissani, M.; Senouci, M.R.; Demigna, W.; Mellouk, A., "*Optimizations and Performance Study of the Dynamic Source Routing Protocol*," Third International Conference on Networking and Services (ICNS'07), June 2007.

[7] C. Perkins, E. Belding-Royer and S. Das, "*Ad hoc On-Demand Distance Vector (AODV) Routing*". RFC 3561, 2003.

[8] Suraj Sharma and Sanjay Kumar Jena. "*A survey on secure hierarchical routing protocols in wireless sensor networks*", In Proceedings of the International Conference on Communication, Computing & Security (ICCCS '11). ACM, New York, NY, USA, 146-151, 2011.

[9] G. Mujica, V. Rosello, J. Portilla, T. Riesgo, "*Hardware-software integration platform for a WSN testbed based on cookies nodes*", in proc. 38th Conference on IEEE Industrial Electronics Society, IECON 2012. pp. 6013-6018.

[10] Liang Jiao; Donghong Qin; Jiahai Yang; Liansheng Ge; Fenglin Qin, "*MRS: A click-based multipath routing simulator*," 7th International ICST Conference on Communications and Networking in China (CHINACOM), 2012, pp.1,6, Aug. 2012.

[11] Gaurav Singh Rawat, Deepak Kumar, and Suresh C. Gupta. "*An analysis of DYMO, FSR, ZRP and RIP routing protocols using QUALNET*". In Proceedings of the International Conference on Interdisciplinary Advances in Applied Computing (ICONIAAC '14). ACM, New York, NY, USA, 2014.

[12] WR. Heinzelman, A. Chandrakasan, H. Balakrishnan. "*Energy-efficient communication protocol for wireless microsensor networks*". Annual Hawaii International Conference on System Sciences, Maui. 2000.

[13] D. Gavalas, G. Pantziou, C. Konstantopoulos, B. Mamalis. "*Lowest-ID with Adaptive ID Reassignment: A Novel Mobile Ad-Hoc Networks Clustering Algorithm*". Proceedings the International Symposium of Wireless Pervasive Computing,. January 2006.

[14] J. Y. Yu, P. H. J. Chong, M. Zhang. "*Performance of Efficient CBRP in Mobile Ad-Hoc Networks (MANETS)*". Vehicular Technology Conference, 2008.

[15] M. Rezaee, M. Yaghmaee. "*Cluster based Routing Protocol for Mobile Ad Hoc Networks*". INFOCOM, Journal of Computer Science, Volume-8, pp.30-36, Feb.2009.

[16] G. Mujica, A. Garcia, J. Gordillo, J. Portilla, T. Riesgo, "*A Novel On-site Deployment, Commissioning and Debugging Technique to Assess and Validate WSN Based Smart Systems*", In proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'15), Lisbon-Portugal, May 2015.